1. Задание

Написать телеграмм бота, который по названию будет искать фильм и предлагать для просмотра ссылки на онлайн кинотеатрах, таких как ivi и okko.

1. Создание бота в telegram

В telegram для создания нового бота нужно через поиск найти бота @BotFather – это отец всех ботов в telegram. С помощью команды /newbot создается новый бот. При создании нужно указать имя бота и его username. После чего @BotFather пришлет смс об успешном создание бота и отправит его token. На Рисунке 1 приведен пример создания бота, где имя – testbot\_alisa, a username – testbot\_alisa\_bot.

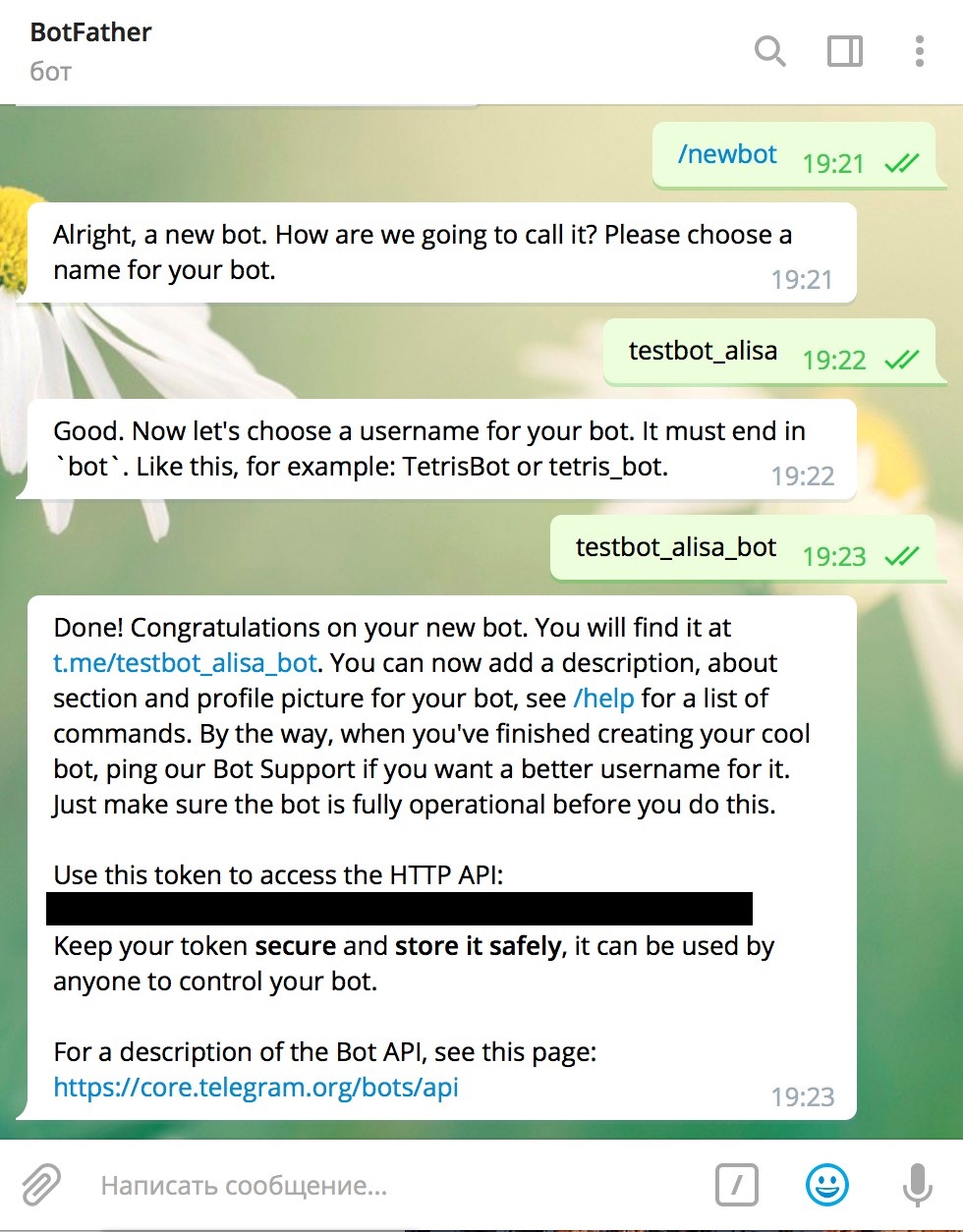


Рисунок 1 – Создание бота.

Важно знать, что token – единственный идентификационный ключ к боту. Это значит, что зная только один токен, можно управлять ботом.

В данной работе для написания бота был выбран язык python 3.7, библиотека aiogram. Аiogram довольно простая и полностью асинхронная библиотека для Telegram API. Инициализировать бота и слушателя можно с помощью следующий команды:

*bot = Bot(token=API\_TOKEN)*

*dp = Dispatcher(bot)*

Где API\_TOKEN – это как раз токен, который выдал бот BotFather.

После создания бота, можно через поиск найти нового бота, но пока что он ничего не может делать.

1. Первый обработчик команды

Для начала работы бота нужно добавить обработчик команды, для вывода приветствия и описания дальнейшего действия. Ниже приведен код, данная функция срабатывает только для команды 'start' и 'help'.

*@dp.message\_handler(commands=['start', 'help'])*

*async def send\_welcome(message: types.Message):*

*"""*

*This handler will be called when client send `/start` or `/help` commands.*

*"""*

*await message.reply("""Hi!*

*I'm CinemaBot!*

*Type the title of film to search it.*

*Привет!*

*Я КиноБот!*

*Набери название фильма, чтобы я его поискал""")*

На Рисунке 2 приведен пример работы команды /start.

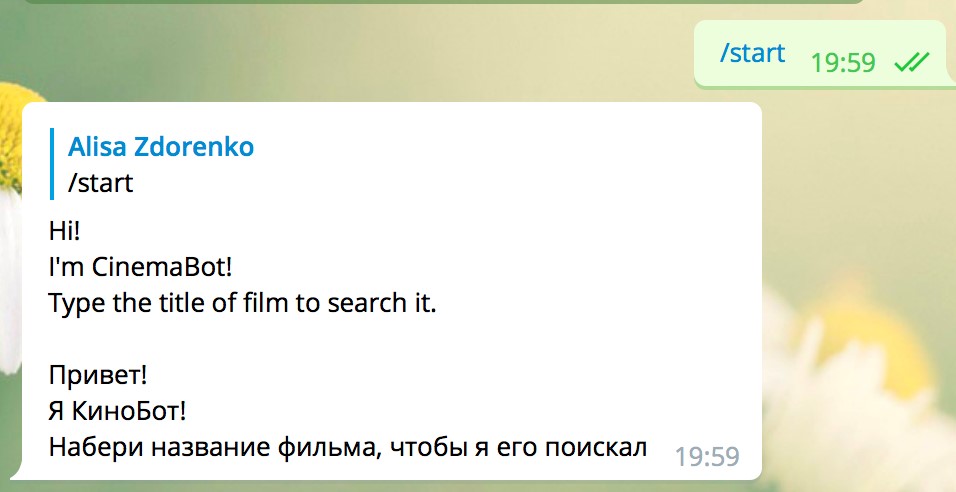


Рисунок 2 – Команда /start

1. Поиск фильма

Бот КиноБот создан для поиска фильмов в онлайн кинотеатрах. Название фильма можно вводить как на русском, так и на английском языке. По результату бот выводит список фильмов. Рассмотрим пример поиска фильма на русском языке. Для этого написана функция russian\_search, которая срабатывает по регулярному выражению: *@dp.message\_handler(regexp='.\*[А-Яа-я].\*').* Если название фильма подходит по регулярному выражению, то отправляется POST запрос по адресу <https://www.kino-teatr.ru>, и с помощью BeautifulSoup происходит разбор ответа. BeautifulSoup поддерживает простые и естественные способы навигации, поиска и модификации дерева синтаксического разбора. Ниже приведен код функции:

*def search\_kinoteatr(film):*

*params = {*

*'text': film.encode('cp1251')*

*}*

*r = requests.post('https://www.kino-teatr.ru/search/', params)*

*soup = BeautifulSoup(r.text, 'lxml-xml')*

*films = []*

*for i, result in enumerate(soup.find\_all('div', 'list\_item\_name')):*

*if i >= 5:*

*break*

*link = result.find('a')*

*film = get\_film\_kinoteatr('https://www.kino-teatr.ru' + link['href'])*

*if film is not None:*

*films.append(film)*

*return films*

На Рисунке 3 приведен пример поиска фильма «Гарри Поттер».

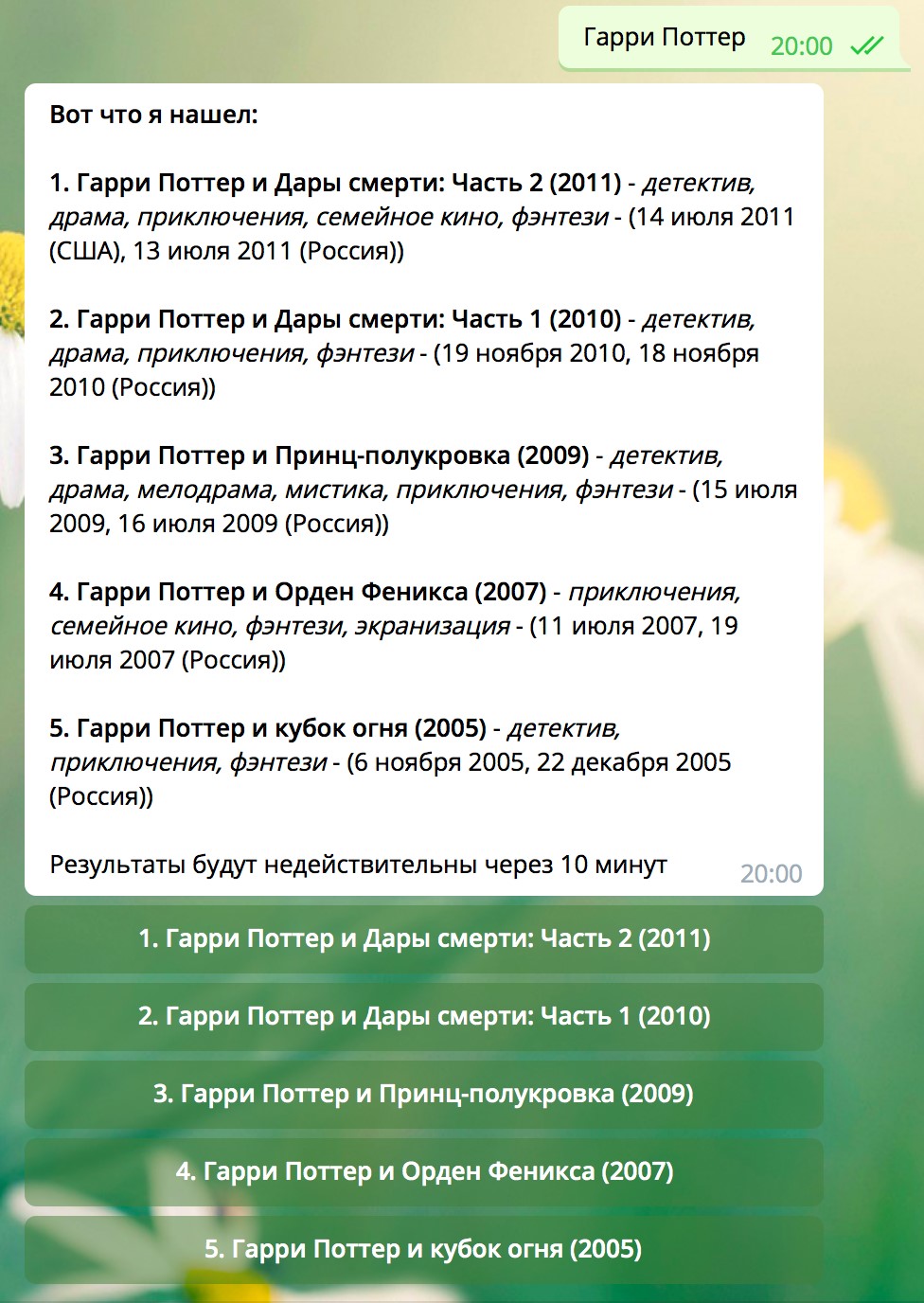


Рисунок 3 – Поиск фильма «Гарри Поттер».

Как видно из рисунка 3 бот нашел 5 фильмов, где встречается название «Гарри Поттер». Для каждого фильма выводится заголовок, жанр, год. Ниже приведен код, который отвечает за вывод списка фильмов.

*def get\_list\_results(films, chat\_id, english=True):*

*markup = types.InlineKeyboardMarkup()*

*if len(films) == 0:*

*msg = 'Sorry, no films with your query.' if english else 'Извините, по вашему запросу ничего не найдено.'*

*else:*

*msg = '<b>The films I have found:</b>\n\n' if english else '<b>Вот что я нашел:</b>\n\n'*

*for i, film\_item in enumerate(films):*

*msg\_item = '<b>{}. {}</b>'.format(i + 1, film\_item.title)*

*markup.add(types.InlineKeyboardButton('{}. {}'.format(i + 1, film\_item.title),*

*callback\_data='{}:{}'.format(i, chat\_id)))*

*if film\_item.genre is not None:*

*msg\_item += ' - <i>{}</i>'.format(film\_item.genre)*

*if film\_item.date is not None:*

*msg\_item += ' - ({})'.format(film\_item.date)*

*msg += '{}\n\n'.format(msg\_item)*

*msg += 'Results will expire in 10 minutes' if english else 'Результаты будут недействительны через 10 минут'*

*return msg, markup*

При клике по фильму из списка, будет выведена более подробная информация о фильме. Ниже приведен код, который отвечает за вывод информации о фильме.

*@dp.callback\_query\_handler(func=lambda query: query.data.split(':')[0] in list(map(str, range(5))))*

*async def film\_info(query: types.CallbackQuery):*

*item, chat\_id = map(int, query.data.split(':'))*

*film\_item = USER\_RESULTS[chat\_id][item]*

*USER\_TIMESTAMPS[chat\_id] = time.time()*

*msg\_item = '<b>{}</b>\n'.format(film\_item.title)*

*if film\_item.genre is not None:*

*msg\_item += '<i>{}</i>\n'.format(film\_item.genre)*

*if film\_item.date is not None:*

*msg\_item += '({})\n\n'.format(film\_item.date)*

*if film\_item.time is not None:*

*msg\_item += '[{}]\n\n'.format(film\_item.time)*

*if film\_item.summary is not None:*

*msg\_item += '{}\n\n'.format(film\_item.summary)*

*for credential, info in film\_item.credits\_dict.items():*

*msg\_item += '<b>{}</b>: {}\n'.format(credential, info)*

*markup = types.InlineKeyboardMarkup()*

*markup.add(types.InlineKeyboardButton('Back',*

*callback\_data='back:{}'.format(chat\_id)))*

*markup.add(types.InlineKeyboardButton('Watch',*

*callback\_data='watch:{}:{}'.format(item, chat\_id)))*

*if film\_item.poster is not None:*

*await bot.send\_message(chat\_id, '[Poster]({})'.format(film\_item.poster), parse\_mode='Markdown')*

*await bot.send\_message(chat\_id, msg\_item, parse\_mode='HTML', reply\_markup=markup)*

На Рисунке 4 приведены пример результата.



Рисунок 4 – Информация о фильме "Гарри Поттер и Принц-полукровка"

После выбора фильма бот предлагает две команды «Назад» и «Смотреть». При клике по кнопке «Назад», будет выведен предыдущий результат поиска. При клике на кнопку «Смотреть» будет выведен список ссылок в онлайн кинотеатрах. Данный бот ищет фильмы только в ivi и okko кинотеатре, пример приведен на Рисунке 5.

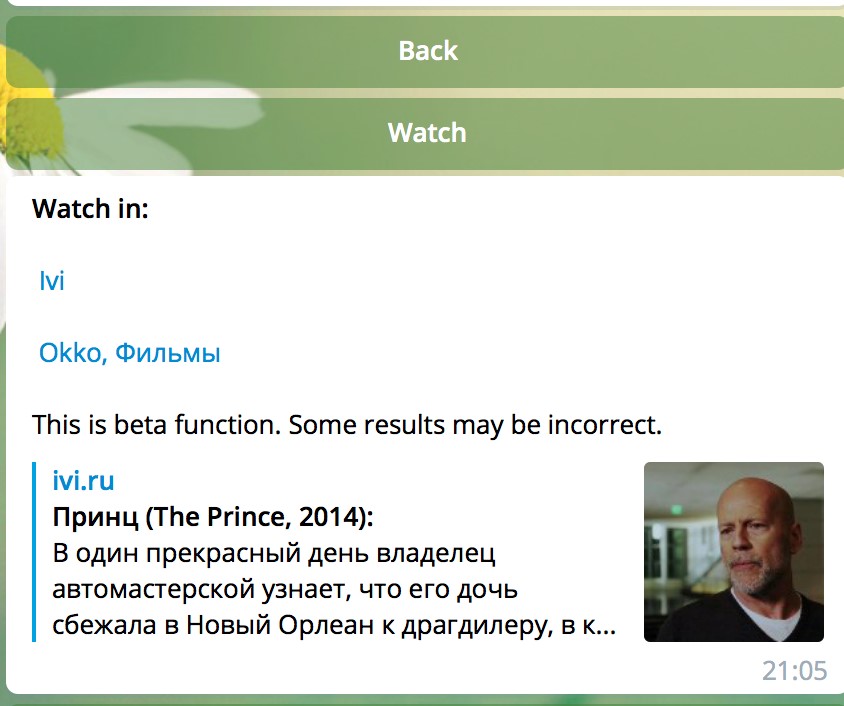


Рисунок 5 – Результат работы команды watch

При клике по ссылке, ссылка отроется в новой вкладке браузера и загрузится нужный фильм. Пример приведен на Рисунке 6.

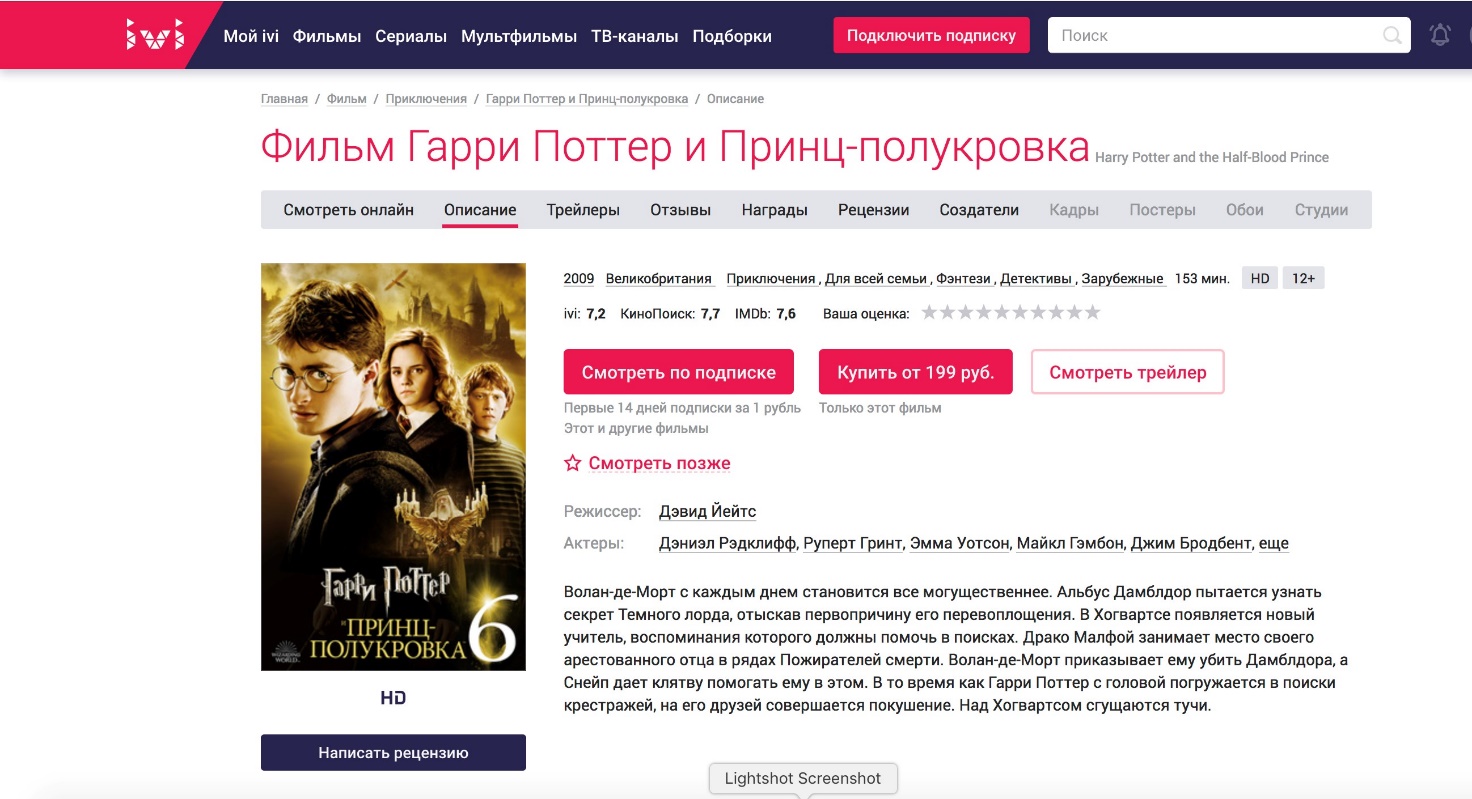


Рисунок 6 – онлайн-кинотеатр ivi